Fall 2014
SEI Research Review
High Confidence Cyber Physical Systems

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Dionisio de Niz
October 28th, 2014

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE
28 OCT 2014

2. REPORT TYPE
N/A

3. DATES COVERED

4. TITLE AND SUBTITLE
HCCPS Line Project Final Review

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
; ; ; Chaki /Dionisio de Niz SagarAndersson /BjornKlein /MarkGurfinkel /Arie

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR’S ACRONYM(S)

11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited.

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**SAR**	**36**	

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM-0001788

Motivation

Many (DoD) systems are Cyber-Physical

- Software tightly coupled with physical world
- Increased scale, complexity, autonomy
  - Pilot Ejection ⇒ IMA ⇒ Multi-UAS Missions

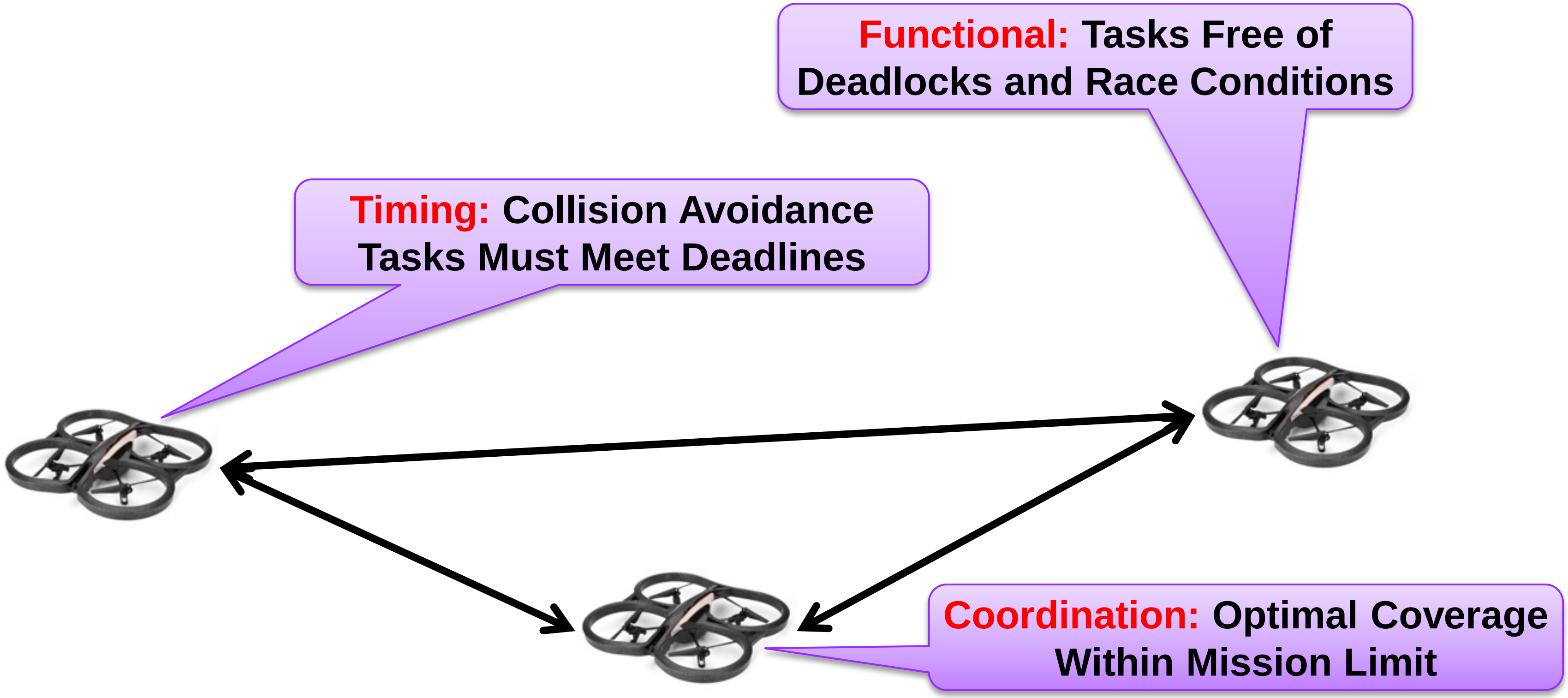
Current DoD T&E regimen is expensive & inadequate to assure CPS

- Testing-based (poor coverage)
- Sufficient assurance needed for Certification

Rigorous assurance of CPS must include at least timing, functionality, and coordination

- Task1 : Timing ⇒ Schedulability analysis: multicore and memory interference
- Task 2: Functional ⇒ Model Checking: scalability, physical laws
- Task 3: Coordination ⇒ Prob. Mod. Checking: compositionality, uncertainty

Guiding Scenario: Multi-UAS Mission



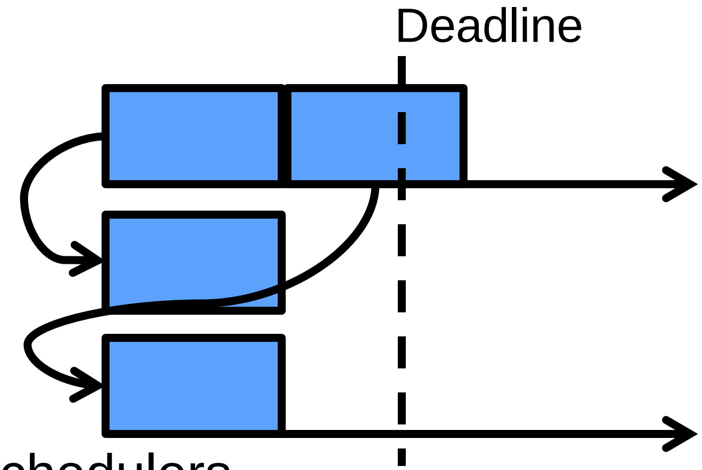
Functional: Tasks Free of Deadlocks and Race Conditions

Timing: Collision Avoidance Tasks Must Meet Deadlines

Coordination: Optimal Coverage Within Mission Limit

Timing, functional correctness, and high-quality coordination are critical to success of modern CPSs. Each must be assured for high confidence in overall performance.

Task 1: Multicore Challenges for Real-Time Systems



Parallelization

- Computation time > Deadline
  - Must parallelized to meet deadline
  - Guarantee always finish before deadline

Shared Hardware Resources / Best Effort Schedulers

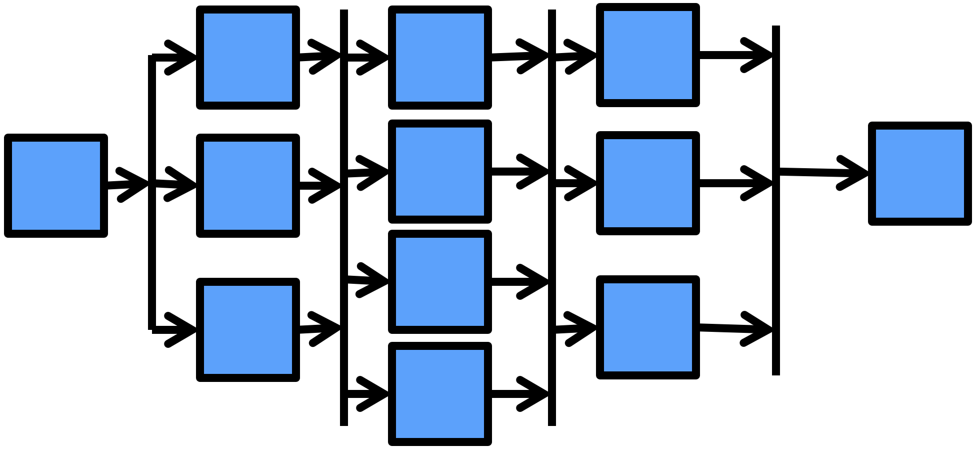
- Shared memory system creates unpredictable delays
- Memory accesses scheduled for average case hinder worst-case

Multiple elements to coordinate

- Shared cache
- Shared main memory
- Shared memory bus

Predictable Parallelization

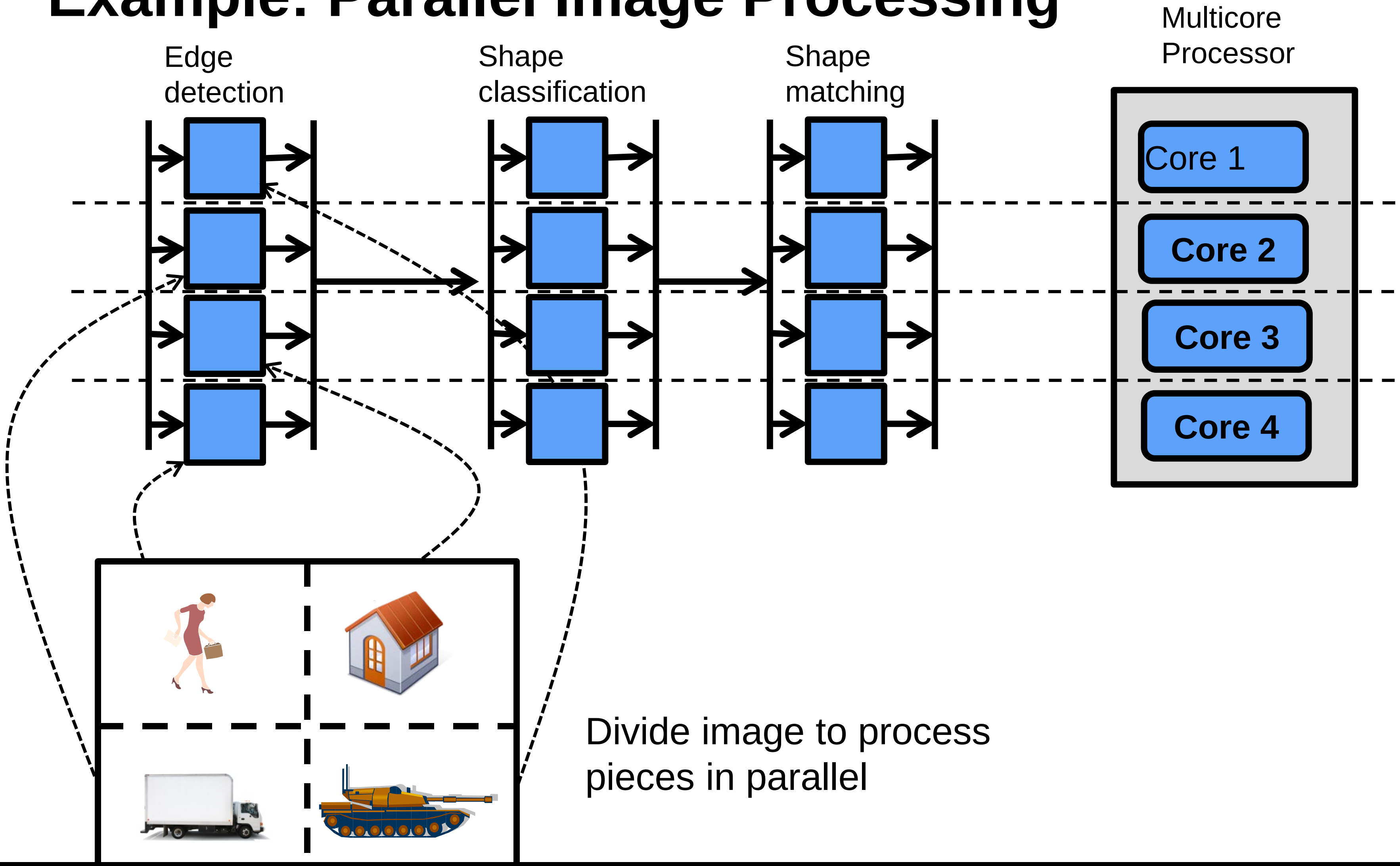
Developed a staged execution model



Scheduled under Global Earliest-Deadline First

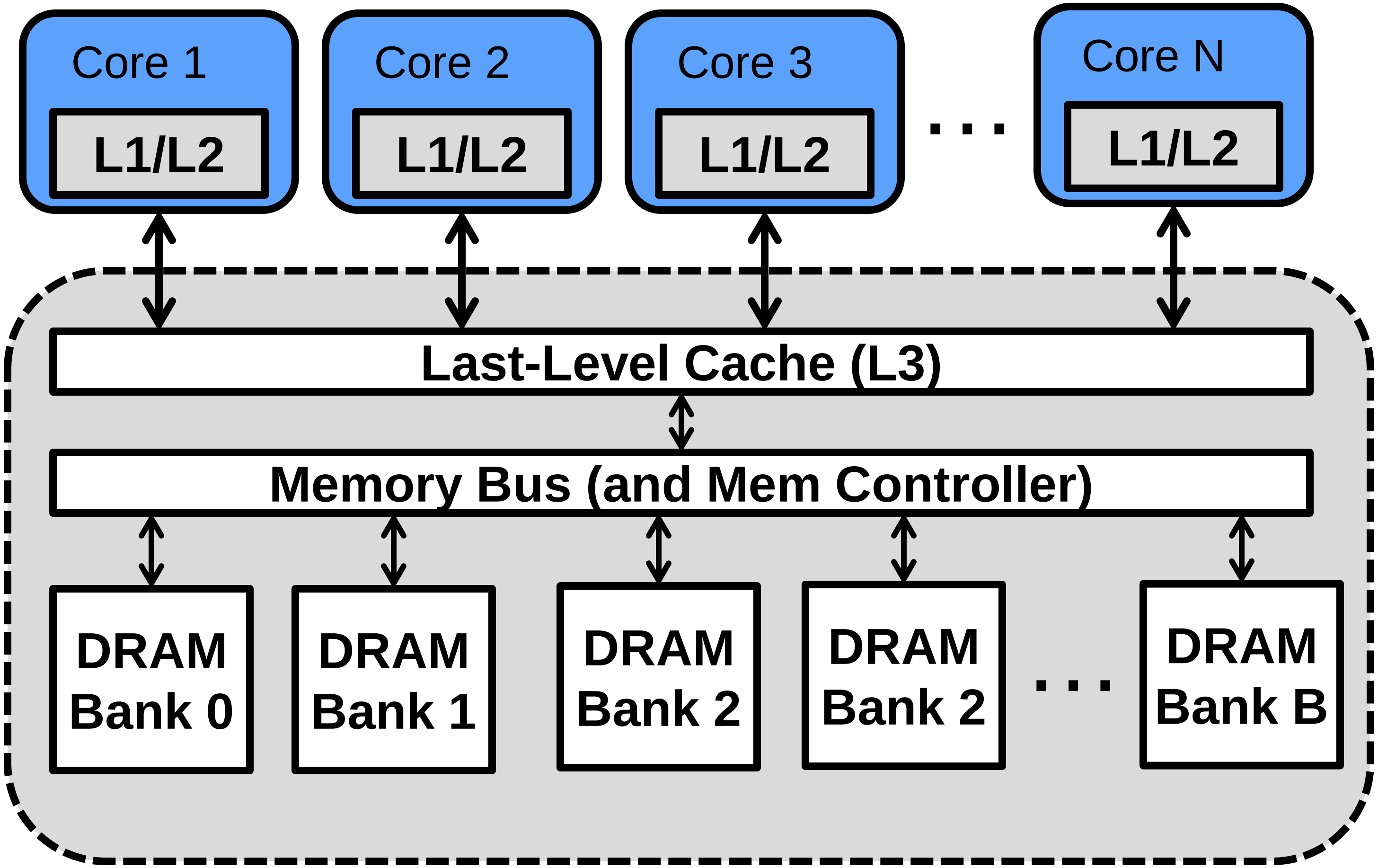
- Most efficient scheduling for staged execution
  - If task schedulable under optimal scheduler our scheduler need at most twice the speed to schedule task

Example: Parallel Image Processing



Divide image to process pieces in parallel

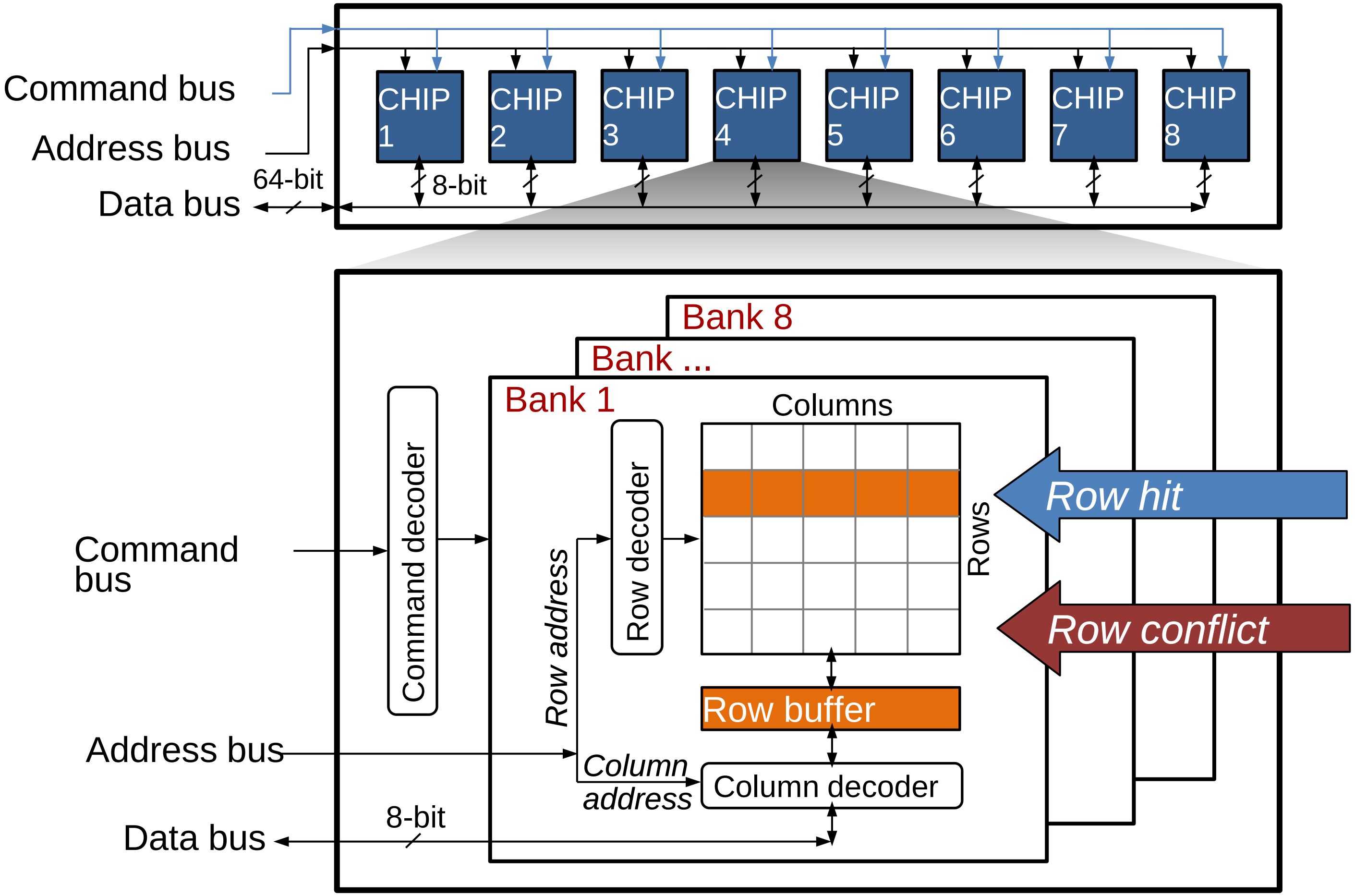
Shared Hardware: Multicore Memory System



DRAM Organization

DRAM Rank

DRAM Chip

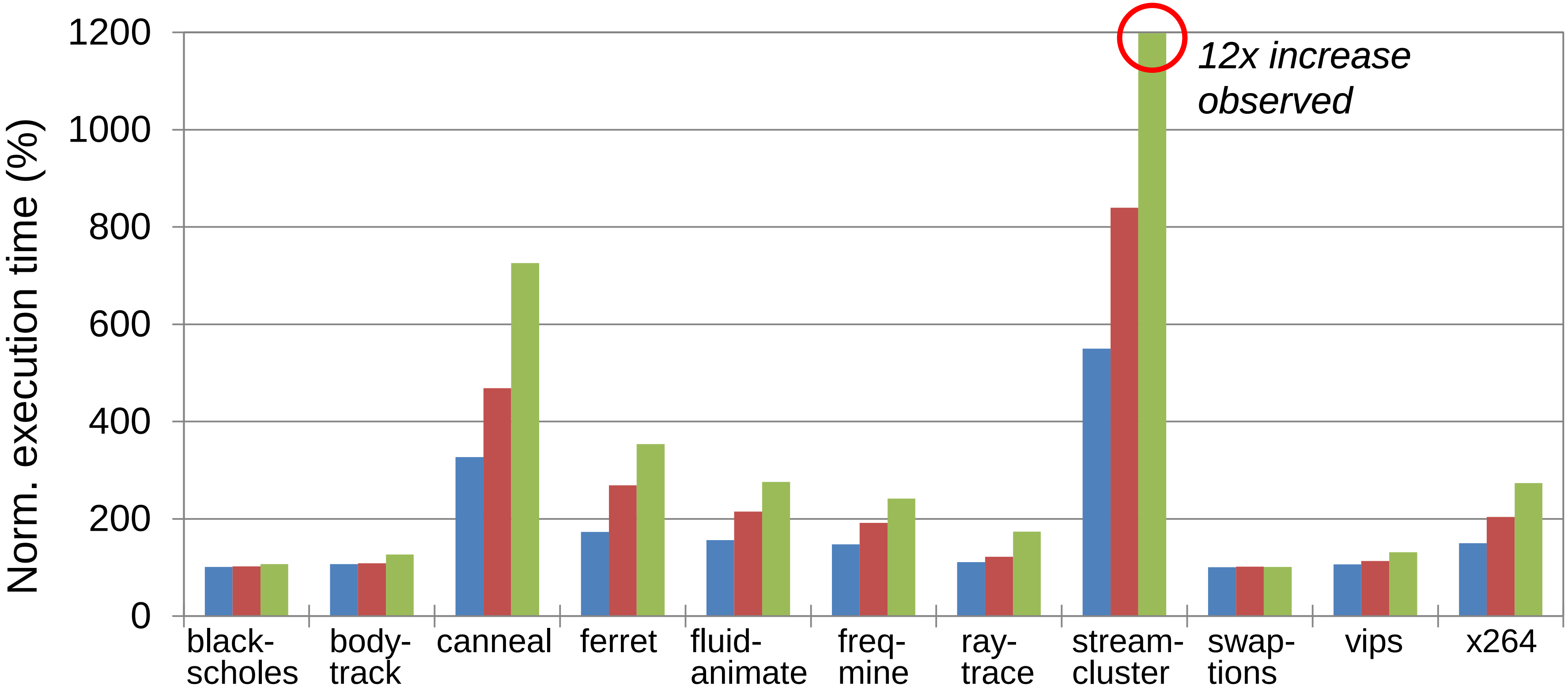


DRAM access latency varies depending on which row is stored in the row buffer

Impact of Memory Interference

- 1 attacker → Max 5.5x increase
- 2 attackers → Max 8.4x increase
- 3 attackers → Max 12x increase

We should predict, bound and reduce the memory interference delay!



12x increase observed

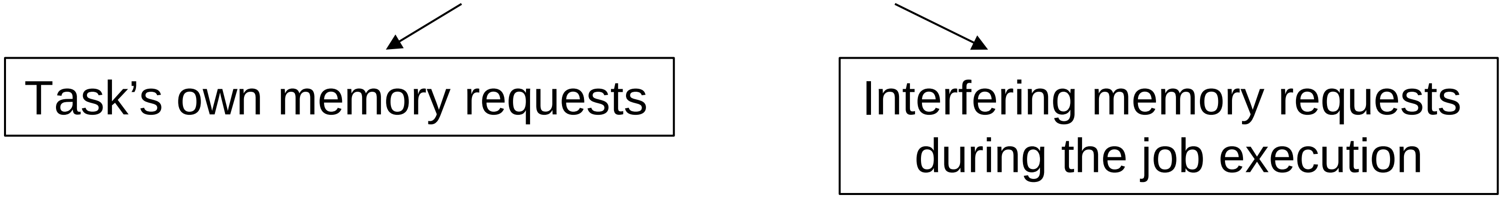
Timing Analysis with Bank Partitions (private/shared)

Explicitly considers the timing characteristics of major DRAM resources

- Rank/bank/bus timing constraints (JEDEC standard)
- Request re-ordering effect

Bounding memory interference delay for a task

- Combines request-driven and job-driven approaches



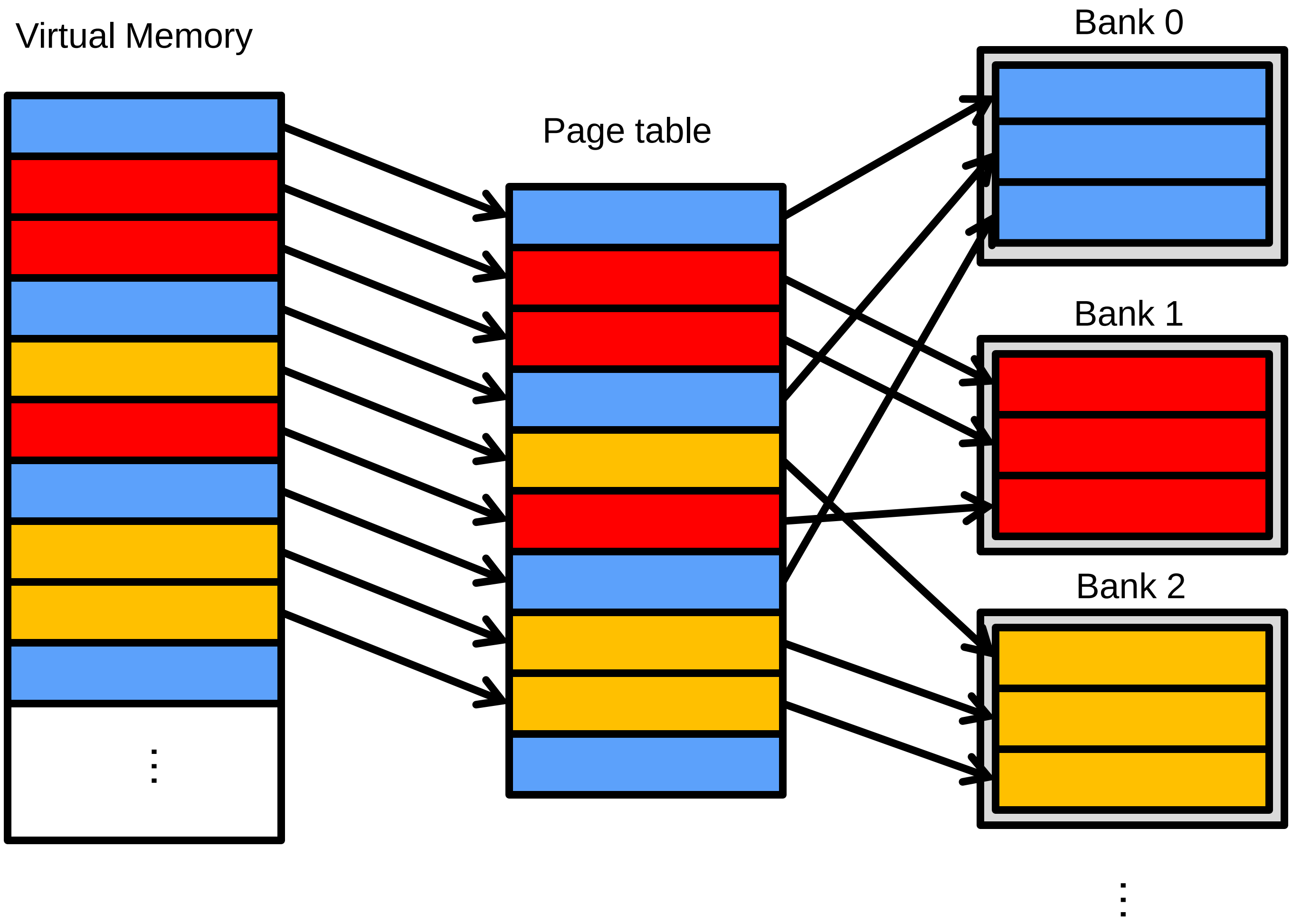
Task's own memory requests

Interfering memory requests during the job execution

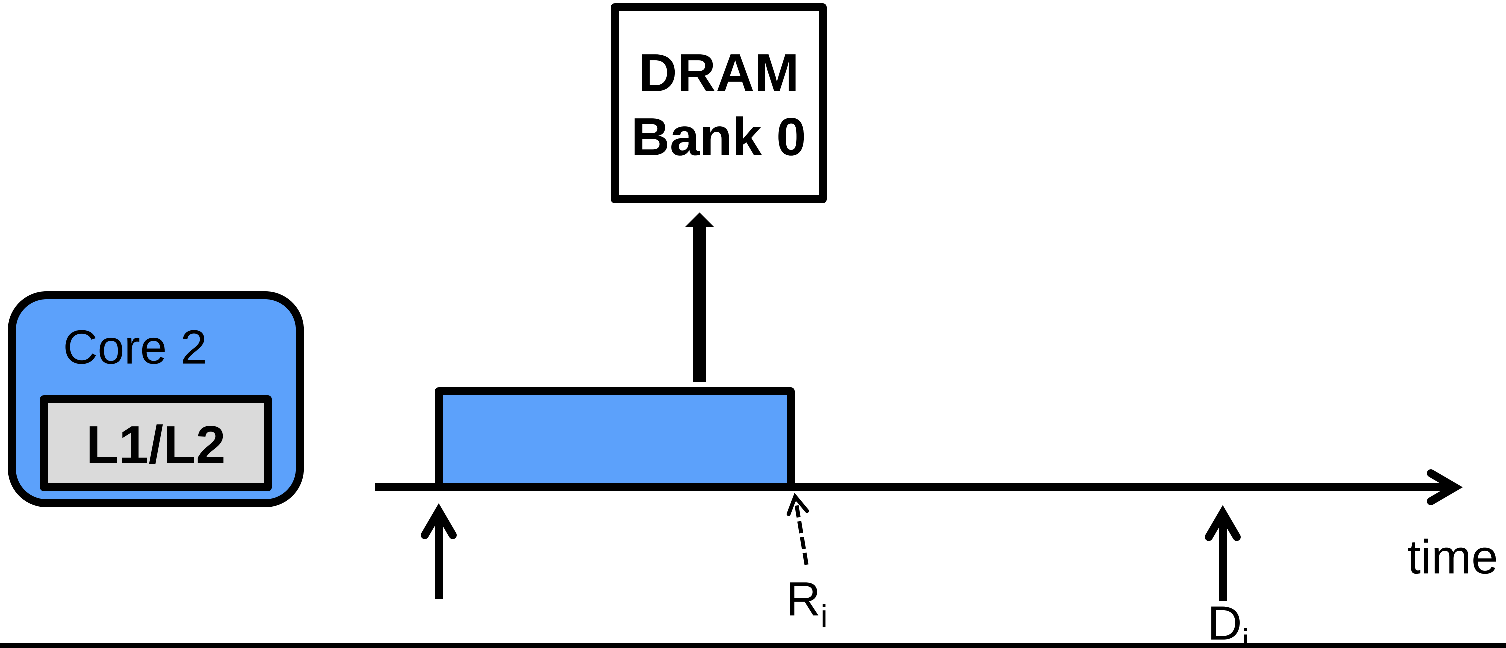
Software **DRAM bank partitioning** awareness

- Analyzes the effect of dedicated and shared DRAM banks

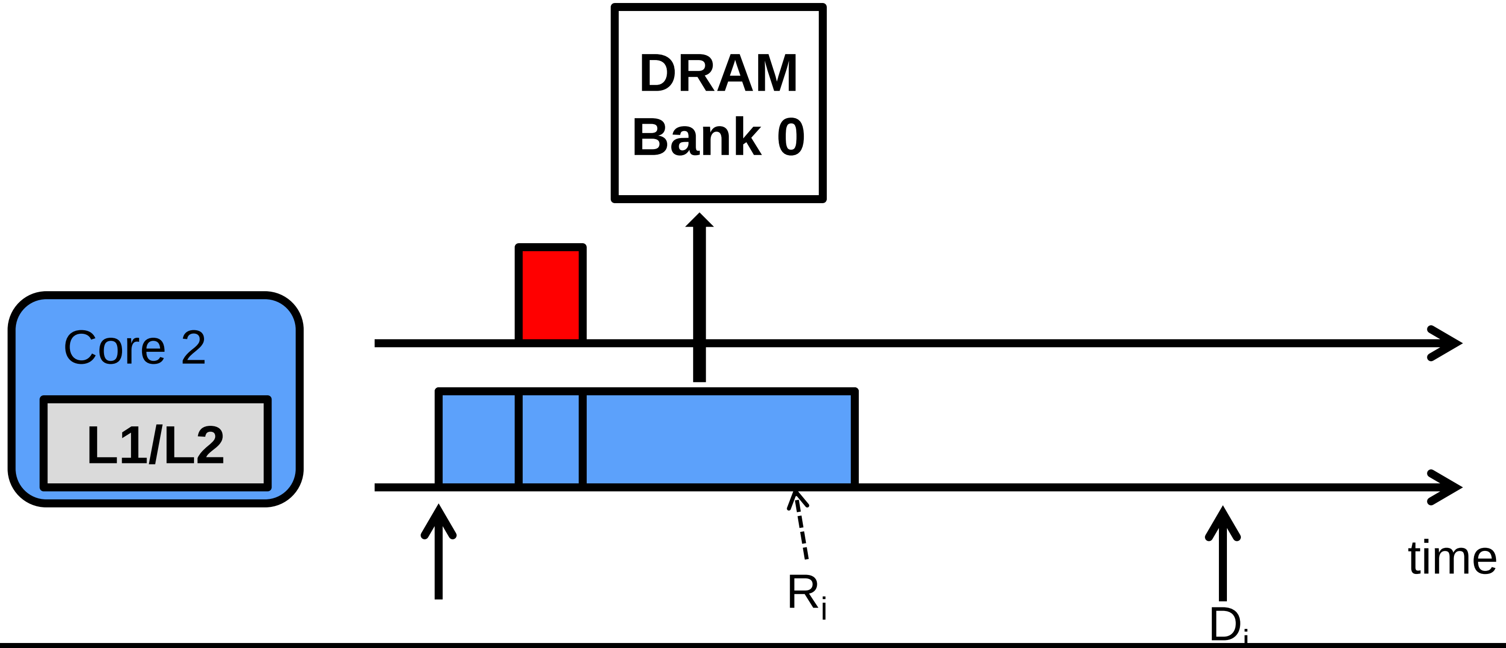
Page Coloring with Virtual Memory



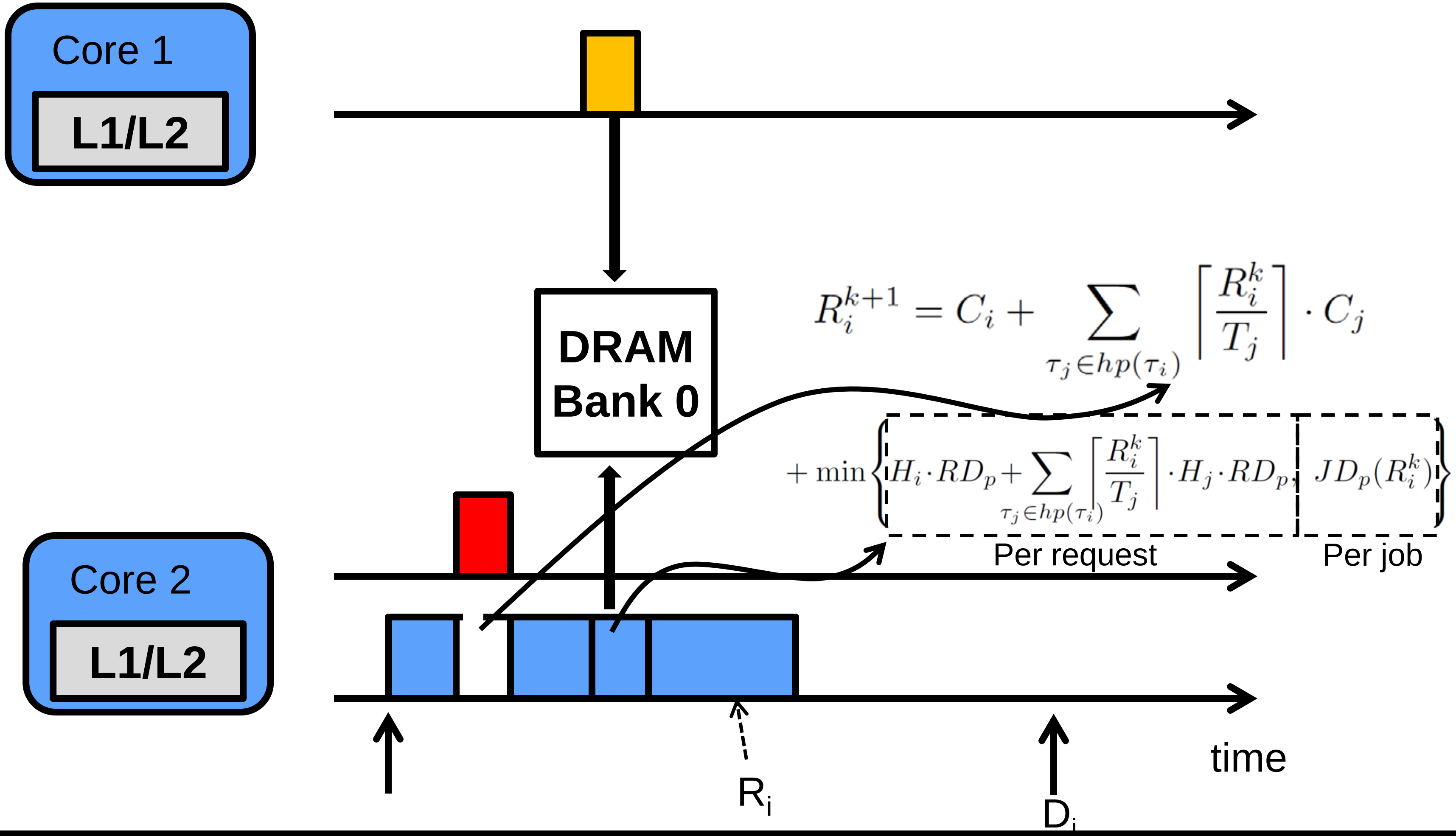
Timing Verification: Response Time(R_i) < Deadline (D_i)



Timing Verification: Response Time(R_i) < Deadline (D_i)



Timing Verification: Response Time(R_i) < Deadline (D_i)

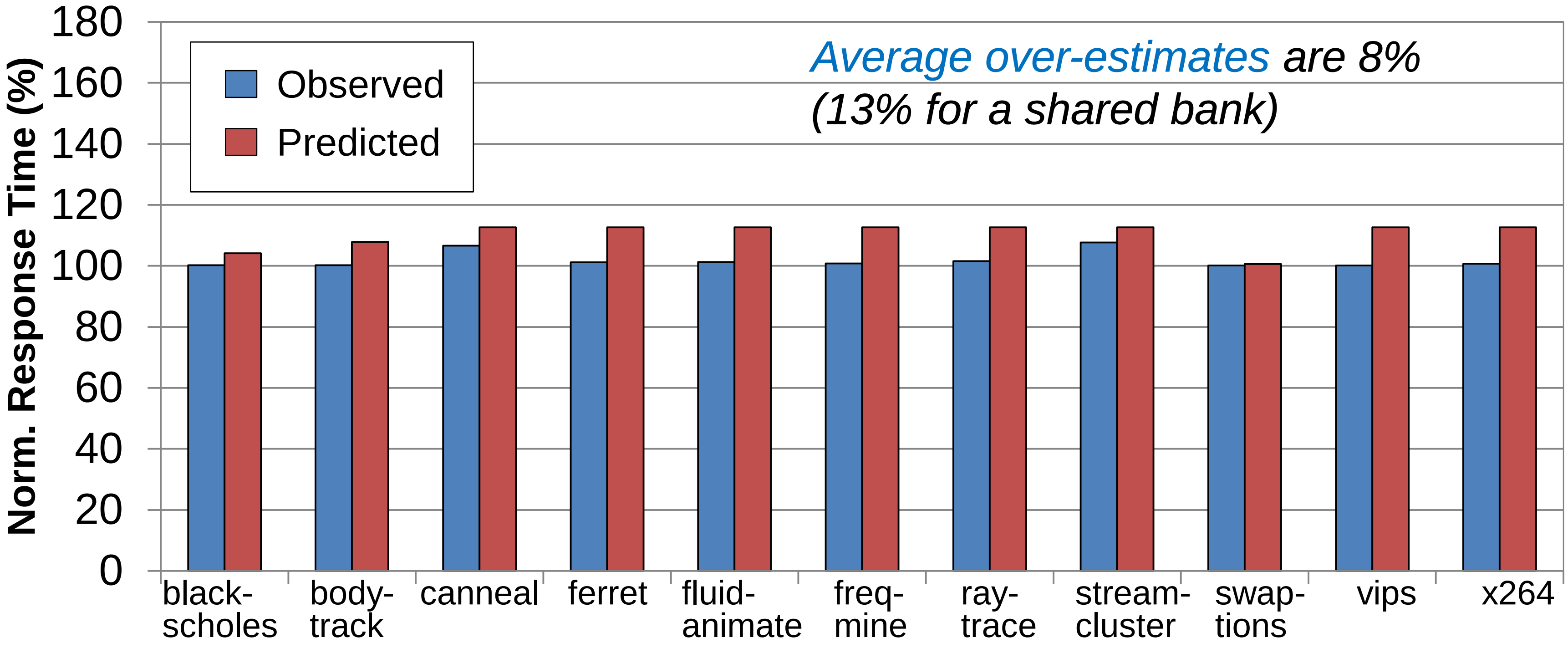


$$R_i^{k+1} = C_i + \sum_{\tau_j \in hp(\tau_i)} \left\lceil \frac{R_i^k}{T_j} \right\rceil \cdot C_j$$

$$+ \min\left\{ H_i \cdot RD_p + \sum_{\tau_j \in hp(\tau_i)} \left\lceil \frac{R_i^k}{T_j} \right\rceil \cdot H_j \cdot RD_p , JD_p(R_i^k) \right\}$$

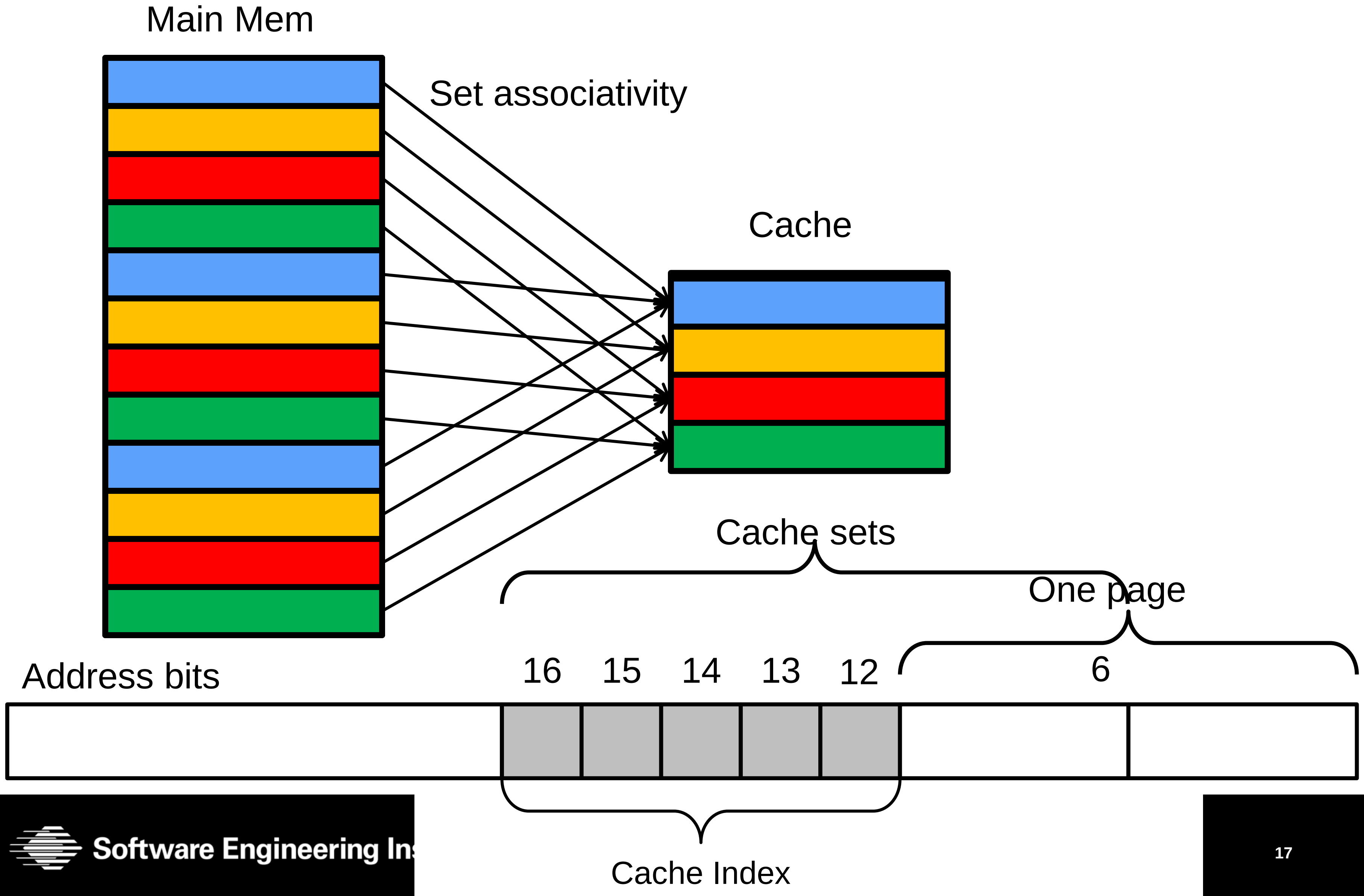
Memory Interference with private banks

- **Private DRAM Bank**

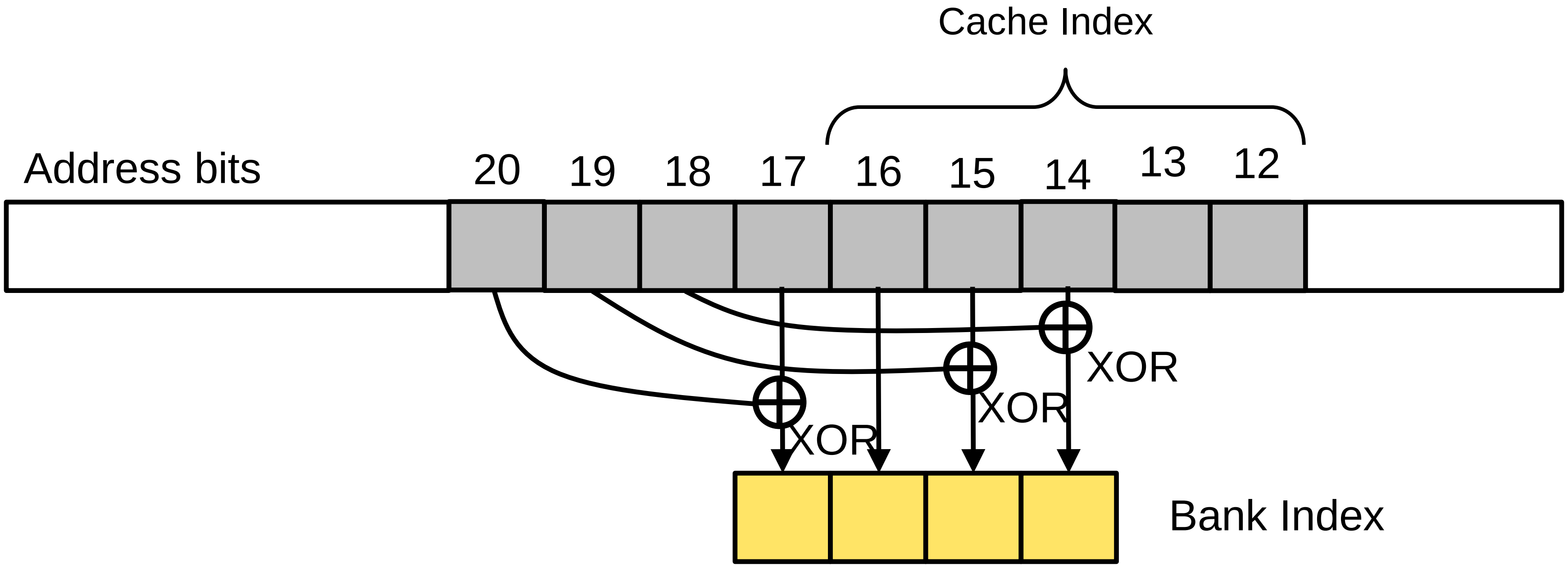


H.Kim, D. de Niz, B. Andersson, M. Klein, O. Mutlu, and R. Rajkumar. "Bounding Memory Interference Delay in COTS-Based Multicore Systems." RTAS 2014. Best Paper.

Cache Partitioning (Coloring)



Cache and Bank Address Bits



E.g. 2 bank bits
2 cache bits
1 shared bit

Bank

Cache	00	01	10	11
00	X		X	
01	X		X	
10		X		X
11		X		X

Coordinated Cache and Bank Partitioning

Avoid conflicting color assignments

Take advantage of different conflict behaviors

- Banks can be shared within same core but not across cores
- Cache cannot be shared within or across cores

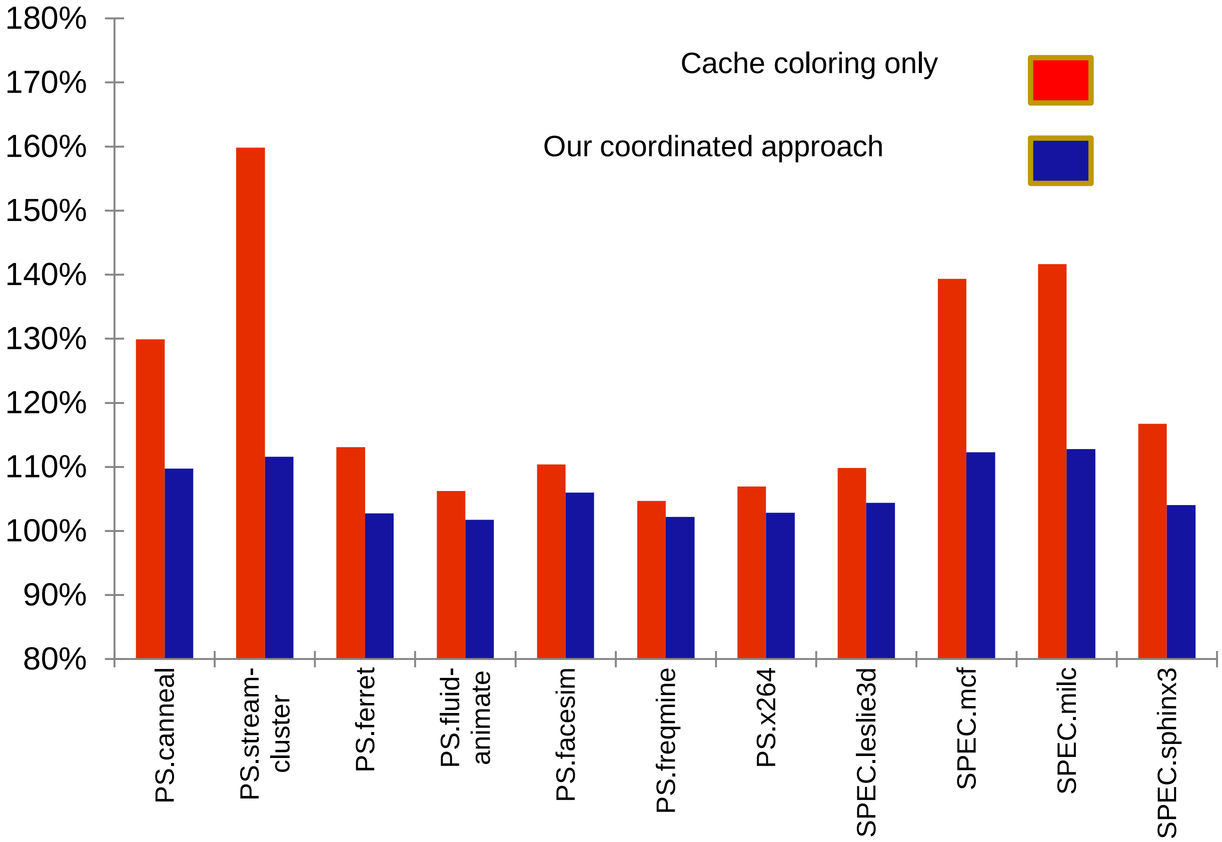
Take advantage of sensitivity of execution time to cache

- Task with highest sensitivity to cache is assigned more cache
- Diminishing returns taken into account

Two algorithms explored

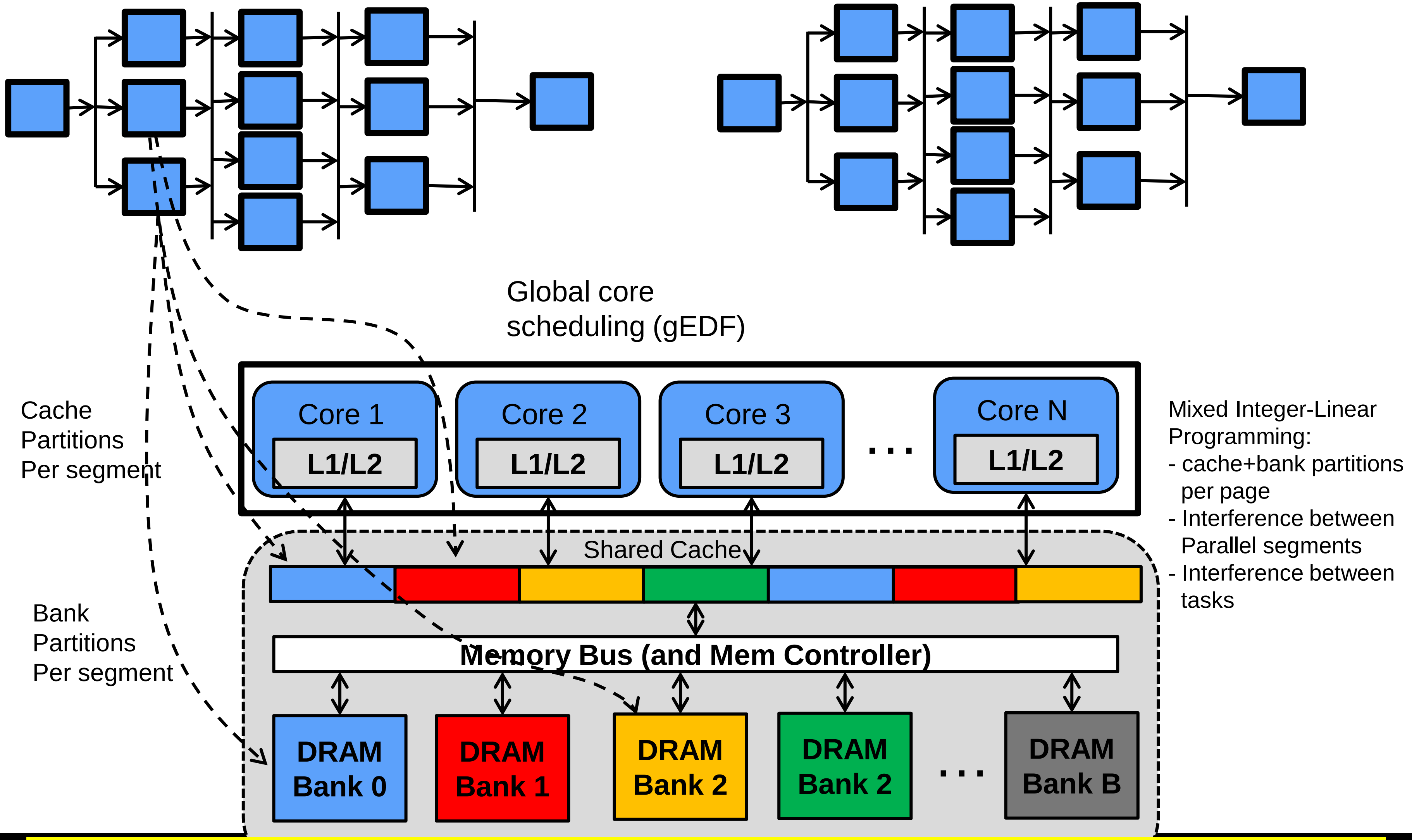
- Mixed-Integer Linear Programming
- Knapsack

Experimental Results



N. Suzuki, H. Kim, D. de Niz, B. Andersson, L. Wrage, M. Klein, and R. Rajkumar. "Coordinated Bank and Cache Coloring for Temporal Protection of Memory Access." ICESS 2013.

Partitions & Scheduling in Parallelized Tasks



Mixed Integer-Linear Programming:

- cache+bank partitions per page
- Interference between Parallel segments
- Interference between tasks

B. Andersson, D. de Niz, H. Kim, M. Klein, and R. Rajkumar. “Scheduling Constrained-Deadline Sporadic Parallel Tasks Considering Memory Contention.” Submitted to: IPDPS 2015.

Round-trip parallelized tasks scheduling

Measure memory accesses per page in a task

- Modified Valgrind profiler to count accesses to a particular virtual page in a program running on the target platform

Assign cache + bank colors to each page and test schedulability

- Mixed-Integer Linear Programming Formulation
- Outputs page per color

Modified Memory System (inside OS) to assign colors per page

- Linux variant (Linux / RK)
- Assign memory reservations (colors) to task and color regions to pages
- Cache + Bank colors

Global Earliest-Deadline First (gEDF) implementation

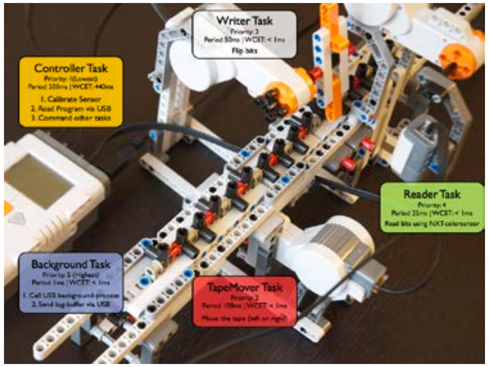
- In Linux / RK

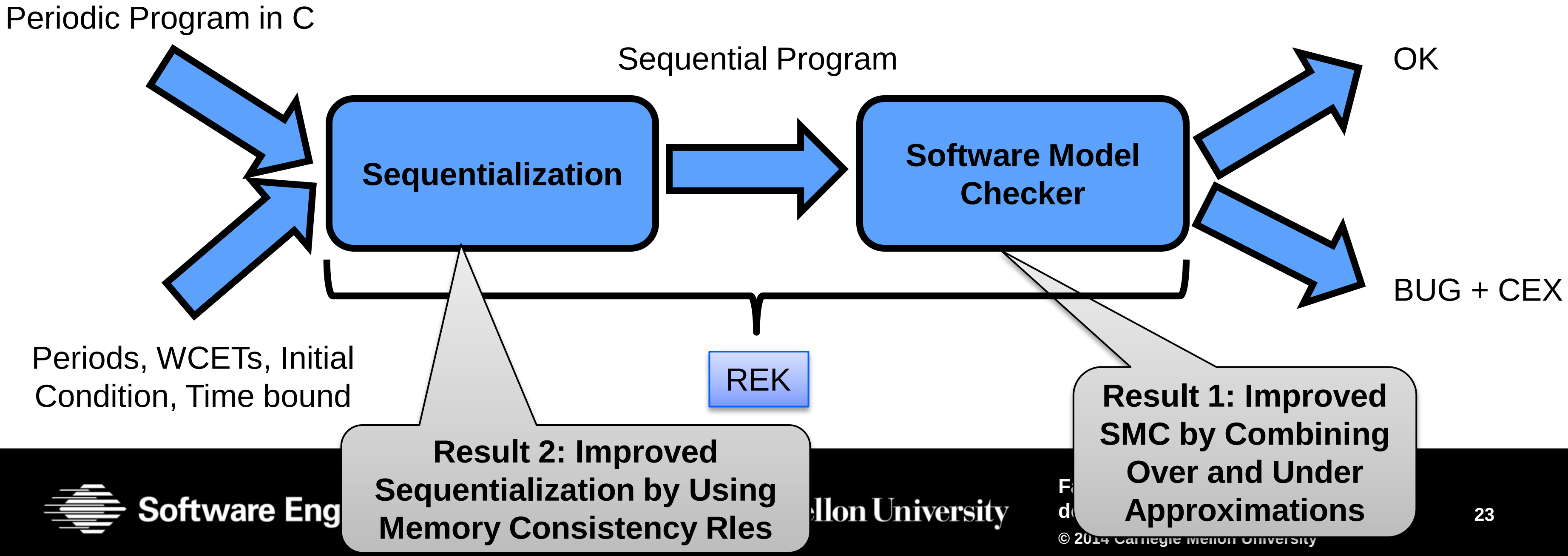
Stage Synchronization Framework

- For Parallel Staged Tasks

Experiments on Intel i7 quad-core 8GB RAM + 8MB Shared Cache

Task 2: Software Model Checking Using Over and Under Approximations





Periodic Program in C

Periods, WCETs, Initial Condition, Time bound

Sequentialization

Sequential Program

Software Model Checker

OK

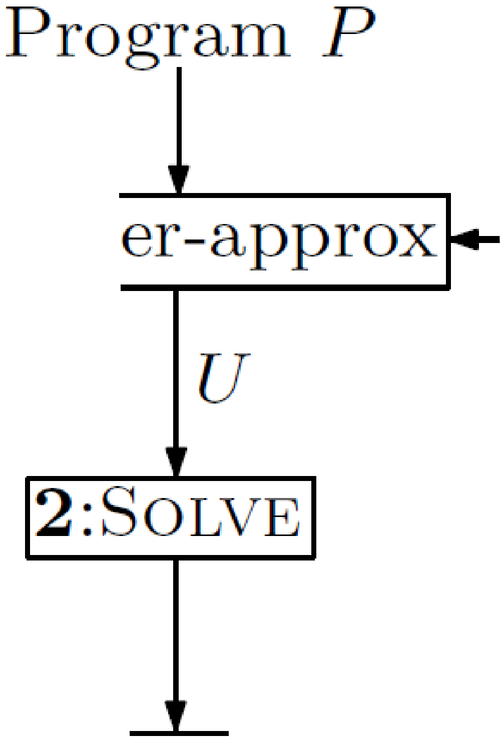
BUG + CEX

REK

Result 2: Improved Sequentialization by Using Memory Consistency Rles

Result 1: Improved SMC by Combining Over and Under Approximations

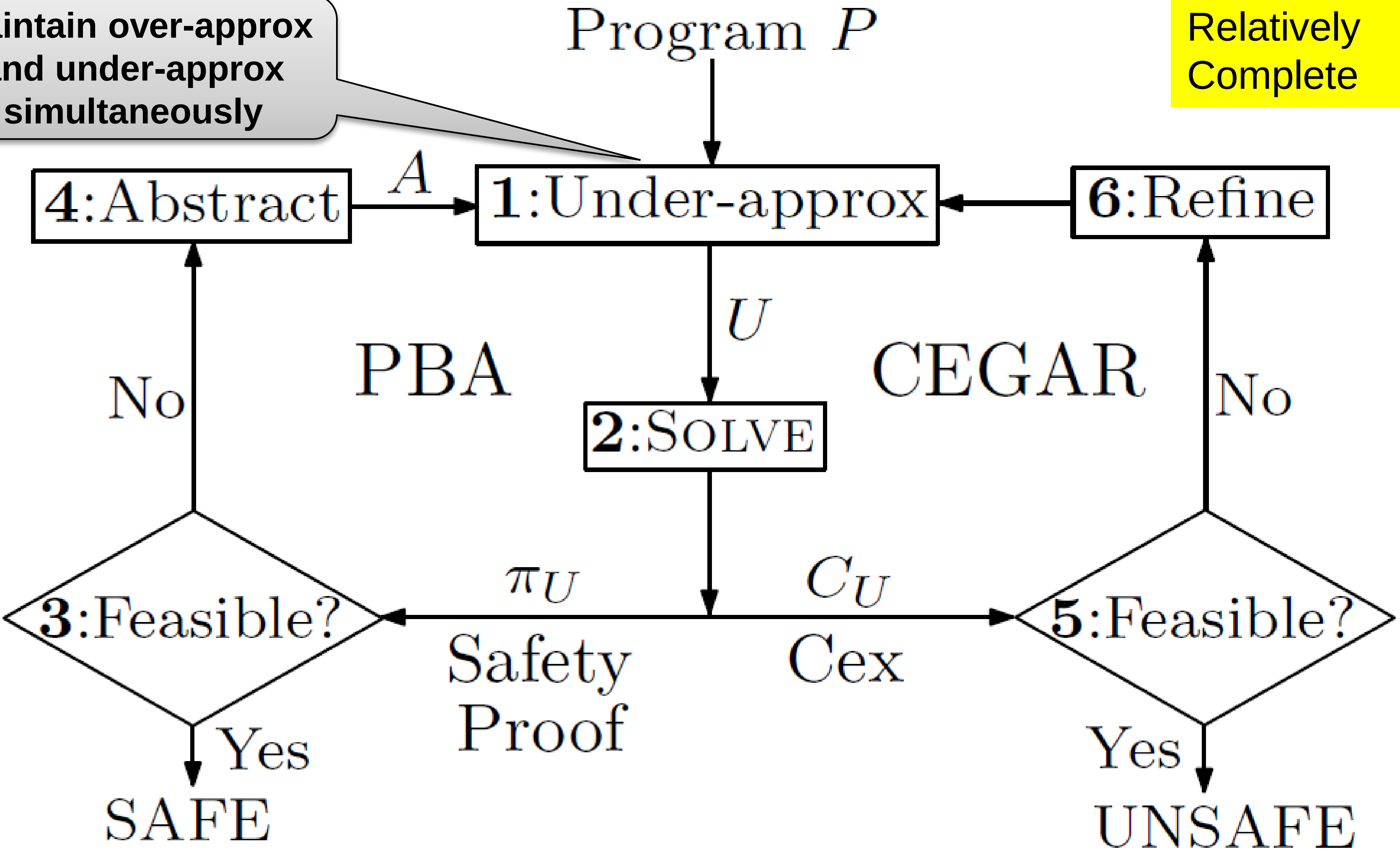
Task 2: Improved Software Model Checking Using Over and Under Approximations



Task 2: Improved Software Model Checking Using Over and Under Approximations

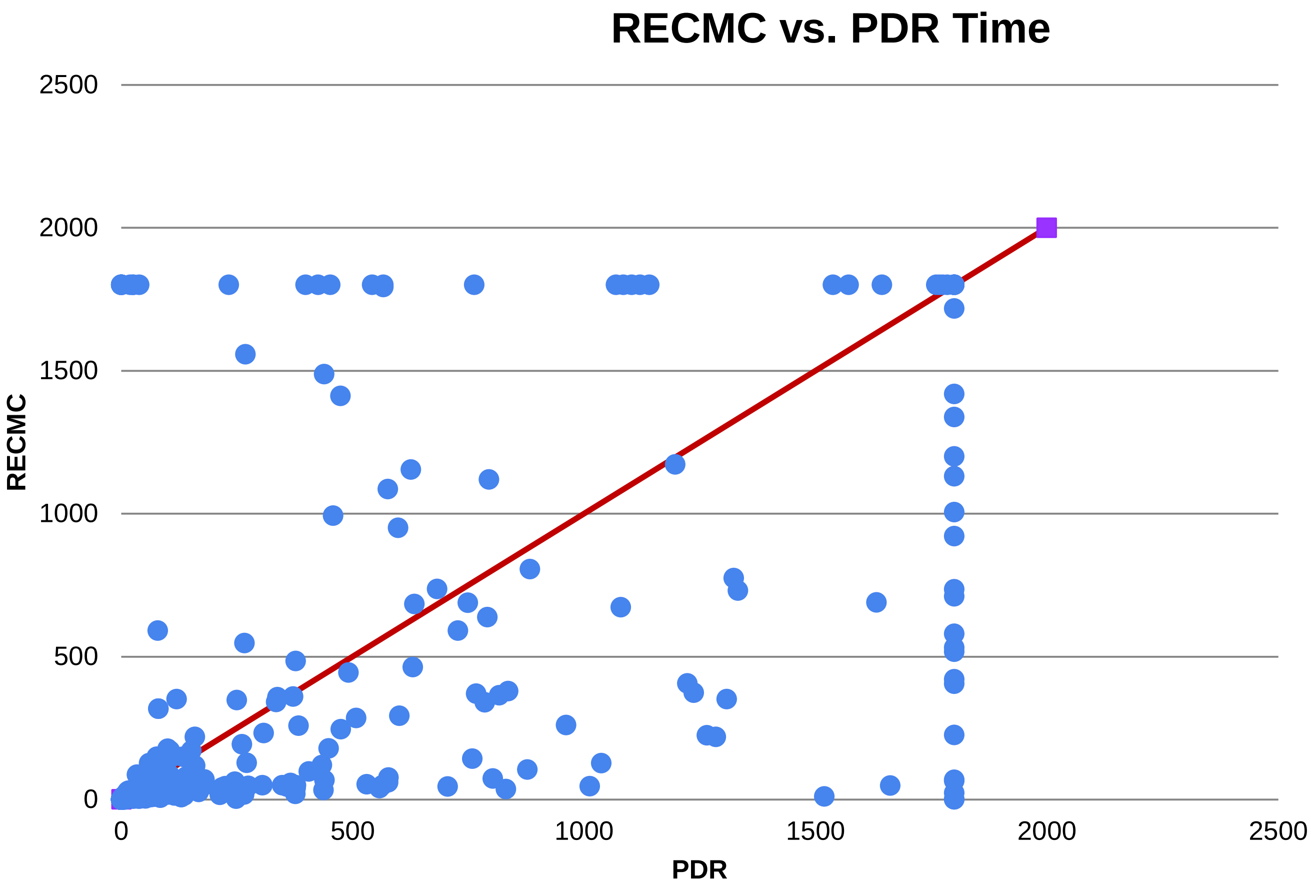
Sound, Relatively Complete

Maintain over-approx and under-approx simultaneously



Publication: Anvesh Komuravelli, Arie Gurfinkel, Sagar Chaki: SMT-Based Model Checking for Recursive Programs. CAV 2014: 17-34

Task 2: Model Checking Results



Software Verification Competition 2014 Benchmarks

Total = 855

RECMC better = 553
PDR better = 232

● Spacer vs. PDR
■ Spacer=PDR

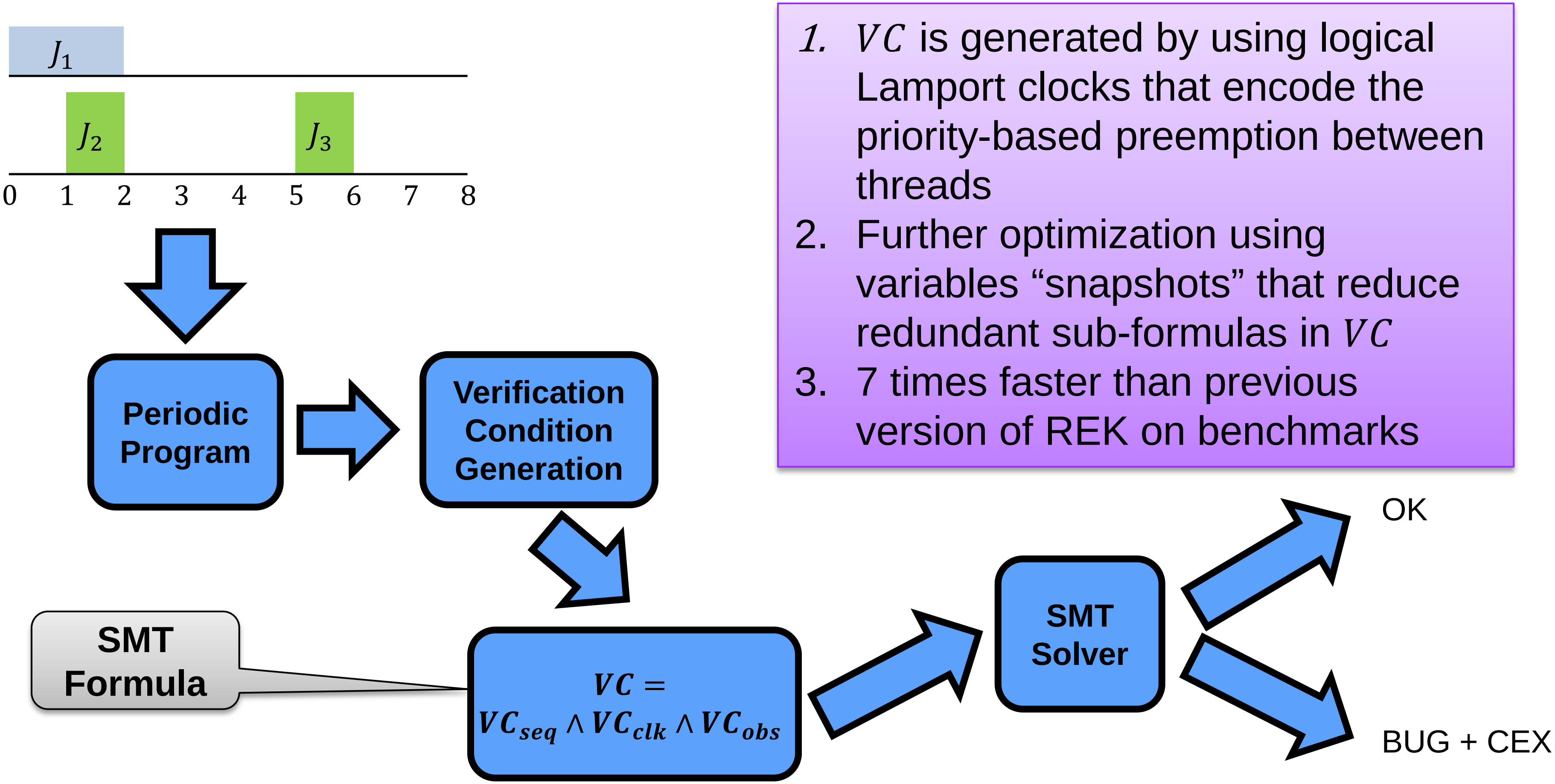
<u>TODO:</u>

Bit-vector semantics

Physical laws : additional theories

PDR = State-of-the-art competitor for RECMC
NOTE: below red line means RECMC better than PDR

Task 2: Improved Sequentialization Using Memory Consistency Rules

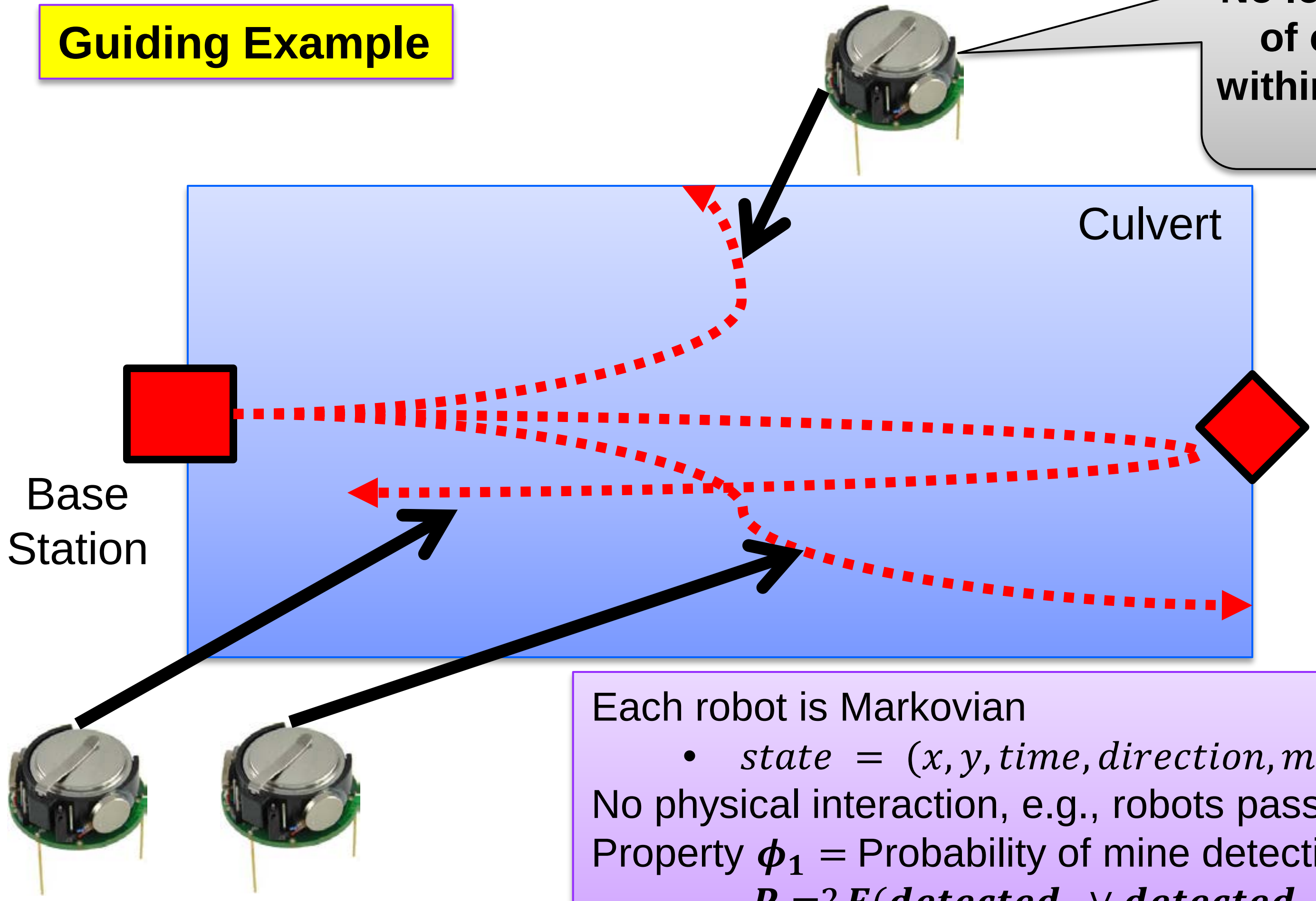

J_1
J_2
J_3
0 1 2 3 4 5 6 7 8
Periodic Program
Verification Condition Generation
SMT Formula
$VC = VC_{seq} \wedge VC_{clk} \wedge VC_{obs}$
SMT Solver
OK
BUG + CEX

1. VC is generated by using logical Lamport clocks that encode the priority-based preemption between threads
2. Further optimization using variables “snapshots” that reduce redundant sub-formulas in VC
3. 7 times faster than previous version of REK on benchmarks

Publication: Sagar Chaki, Arie Gurfinkel, Nishant Sinha: Efficient Verification of Periodic Programs Using Sequential Consistency and Snapshots. FMCAD 2014

Task 3: Probabilistic Model Checking to evaluate Coordinated Multi-Robot Missions

Guiding Example



No localization, disc model of communication (i.e., within radius), probabilistic movement

Base station and Mine have disc model of communication

Each robot is Markovian

- $state = (x, y, time, direction, mine\_detected)$.

No physical interaction, e.g., robots pass through

Property $\boldsymbol{\phi_1}$ = Probability of mine detection.

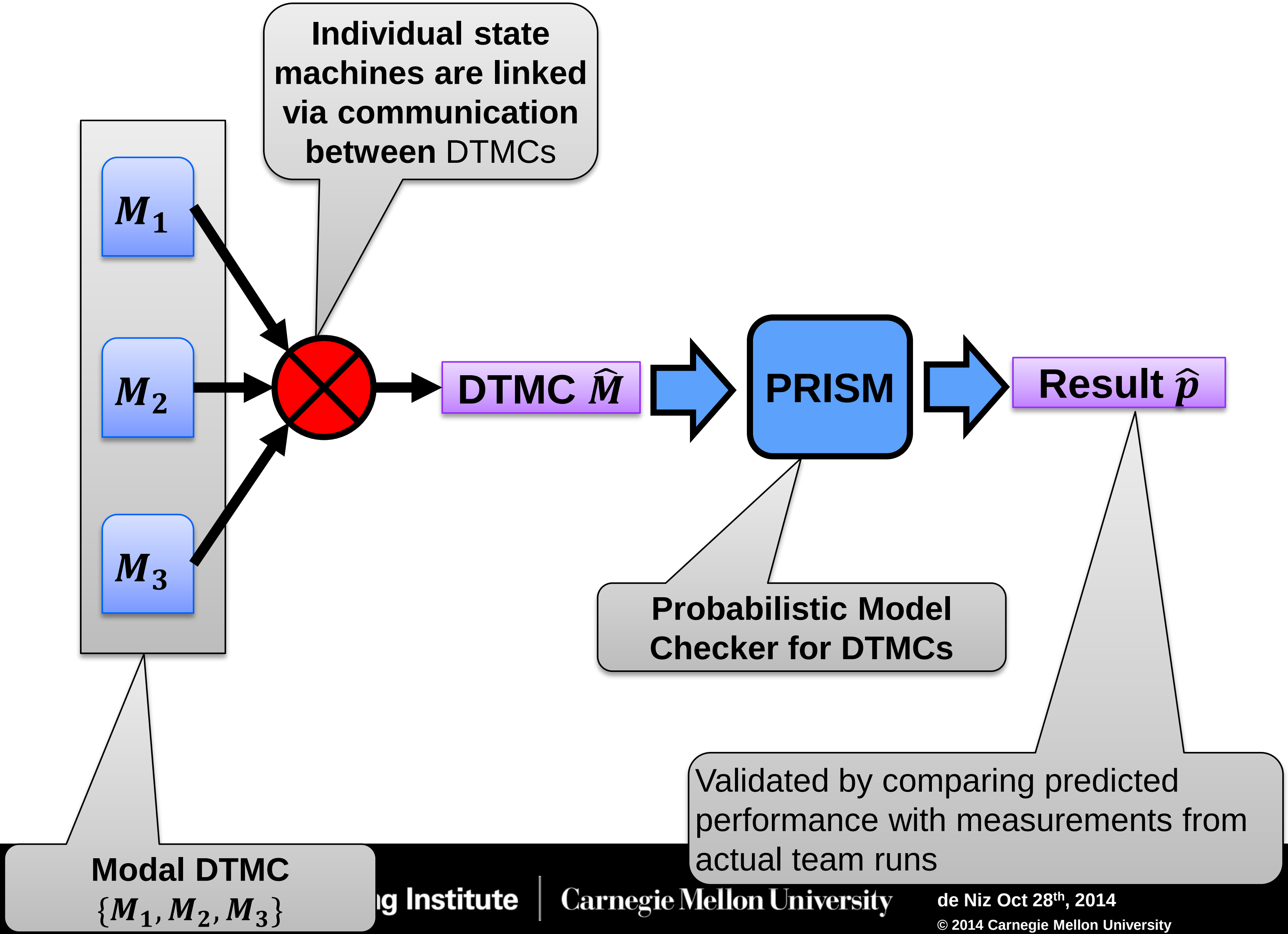
$$\boldsymbol{P = ?\, F(detected_1 \lor detected_2 \lor detected_3)}$$

Property $\boldsymbol{\phi_2}$ = Probability of detection and return to base.

$$\boldsymbol{P = ?\, F(detected_1 \land dir_1 = back \land x_1 = 0 \land y_1 = 1 \ldots)}$$

Property $\boldsymbol{\phi_3}$ = Expected number of robots returning to base.

Overall Approach



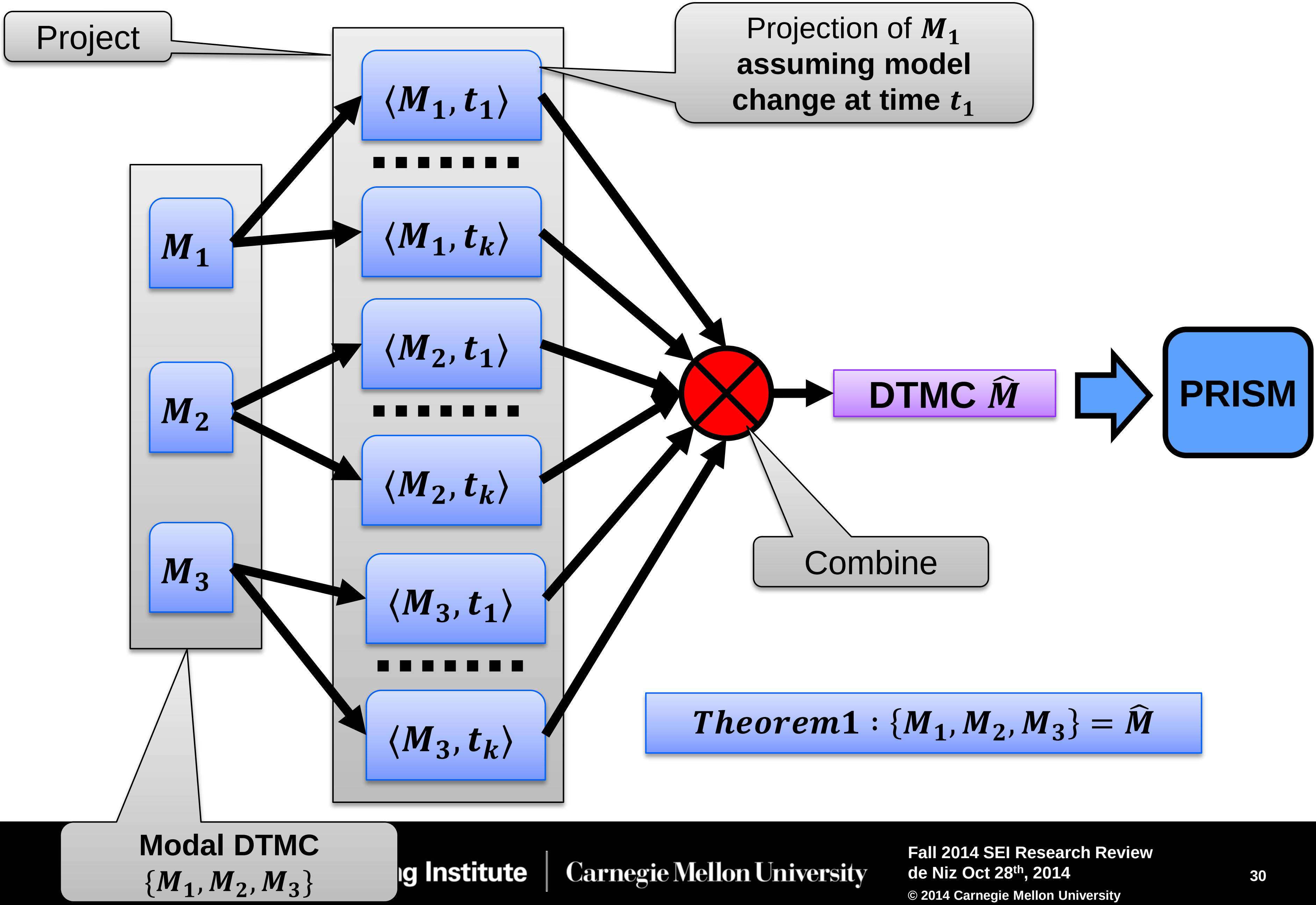
Individual state machines are linked via communication between DTMCs

Probabilistic Model Checker for DTMCs

Validated by comparing predicted performance with measurements from actual team runs

Modal DTMC
$\{M_1, M_2, M_3\}$

Technical Details


Project
Projection of M_1 assuming model change at time t_1
M_1
M_2
M_3
$\langle M_1, t_1 \rangle$
$\langle M_1, t_k \rangle$
$\langle M_2, t_1 \rangle$
$\langle M_2, t_k \rangle$
$\langle M_3, t_1 \rangle$
$\langle M_3, t_k \rangle$
DTMC $\widehat{M}$
PRISM
Combine
Modal DTMC $\{M_1, M_2, M_3\}$

$$Theorem1 : \{M_1, M_2, M_3\} = \widehat{M}$$

Computing $\langle \boldsymbol{M_i}, \boldsymbol{t_j} \rangle$

Physically run Kilobot R_i and force it to turn around at t_j

- Discretize time and space
- Reprogram controller to “fake” mine detection at time t_j
- Transition probability matrix of $\langle \boldsymbol{M_i}, \boldsymbol{t_j} \rangle$ is defined as:
  - $P(s, s') = \frac{n(s,s')}{n(s)}$
  - $n(s)$ = no. of times robot was in state s
  - $n(s, s')$ = no. of times robot moved from s to s' in one time step

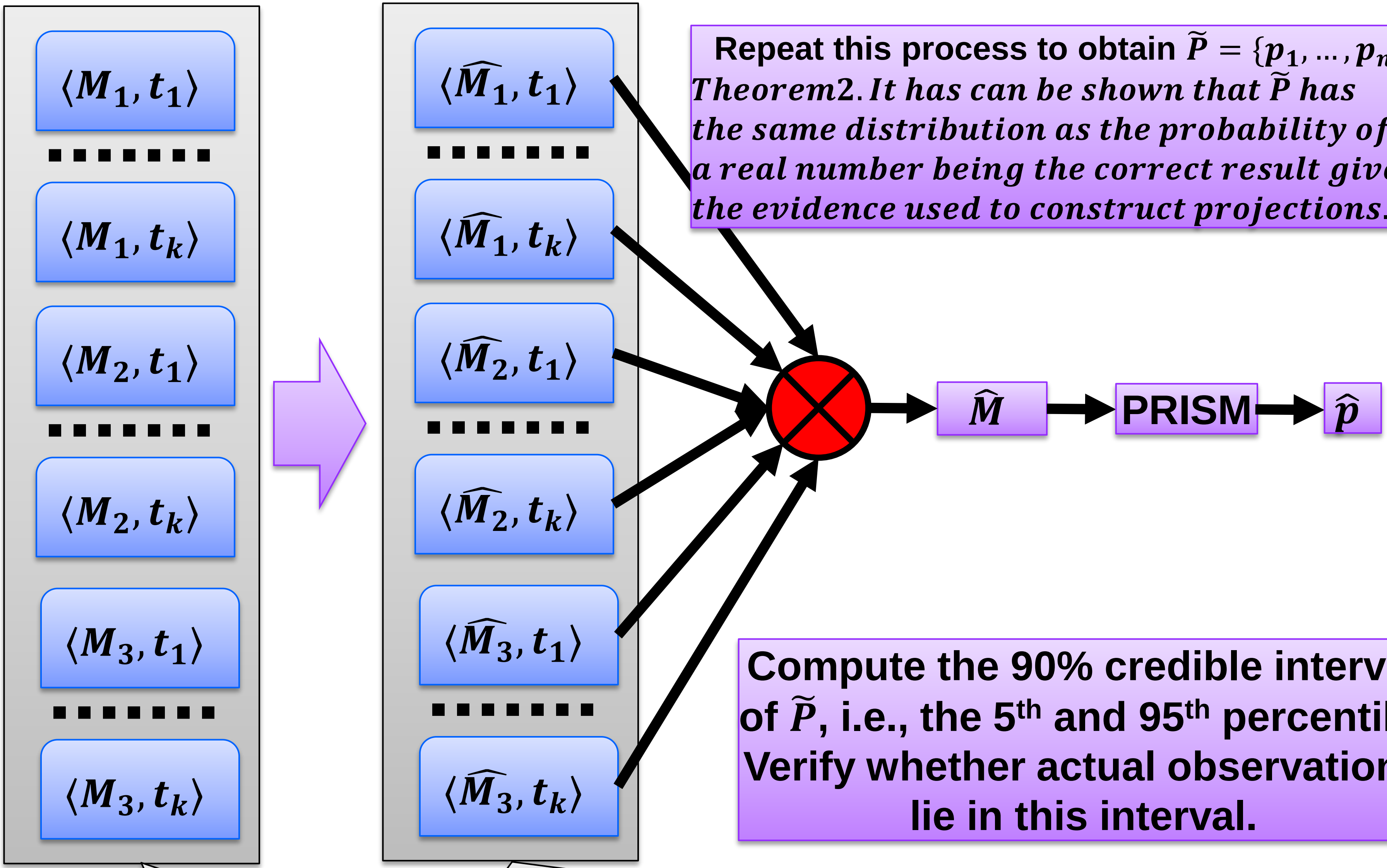
Tedious to repeat these experiments using actual Kilobots

- Use a simulator (VREP)
- Tune parameters to reproduce behavior observed with real Kilobots

At least two sources of error

- Finite number of observations & space and time discretization
- Both will remain no matter how much effort we put in
- How do we quantify and bound the error?

Error Quantification: Fuzzy Sampling


$\langle M_1, t_1\rangle$
$\langle M_1, t_k\rangle$
$\langle M_2, t_1\rangle$
$\langle M_2, t_k\rangle$
$\langle M_3, t_1\rangle$
$\langle M_3, t_k\rangle$
$\langle \widehat{M_1}, t_1\rangle$
$\langle \widehat{M_1}, t_k\rangle$
$\langle \widehat{M_2}, t_1\rangle$
$\langle \widehat{M_2}, t_k\rangle$
$\langle \widehat{M_3}, t_1\rangle$
$\langle \widehat{M_3}, t_k\rangle$
$\widehat{M}$
PRISM
$\widehat{p}$

Repeat this process to obtain $\widetilde{P} = \{p_1, \dots, p_n\}$ *Theorem2. It has can be shown that* $\widetilde{P}$ *has the same distribution as the probability of a real number being the correct result given the evidence used to construct projections.*

Compute the 90% credible interval of $\widetilde{P}$, i.e., the 5th and 95th percentile. Verify whether actual observations lie in this interval.

Projections Constructed P

Perturbed **Projection Constructed using Dirichlet distributions with parameter** P

Results: Probability that one Robot detected the mine and returned to the base = Success

Team in Release Order	Observed	Predicted	Sample Mean	Sample 5%	Sample 95%	
3-2-1	1	0.96	0.96	0.91	0.99	★
4-6-1	0.97	0.96	0.96	0.91	0.99	
4-6-2	0.47	0.43	0.43	0.29	0.58	
5-6-2	0.5	0.43	0.43	0.28	0.61	
5-6-7	0	0	0	0	0	
6-1-7	0.93	0.96	0.96	0.91	0.99	
6-5-7	0	0	0	0	0	
7-3-5	0.7	0.83	0.83	0.72	0.92	★
7-3-6	0.83	0.83	0.84	0.74	0.92	
7-6-1	0.9	0.96	0.96	0.92	0.99	★

Each projection constructed using 30 simulations

Results: Expected Number of Robots that Returned to the Base

Team in Release Order	Observed	Predicted Oneshot	Sample Mean	Sample 5%	Sample 95%	
3-2-1	2.2	2.17	2.17	1.97	2.38	
4-6-1	1.67	1.23	1.23	1.14	1.33	★
4-6-2	0.83	0.7	0.7	0.55	0.89	
5-6-2	0.83	0.72	0.73	0.53	0.91	
5-6-7	0.43	0.29	0.29	0.19	0.38	★
6-1-7	1.57	1.23	1.24	1.14	1.35	★
6-5-7	0.2	0.29	0.3	0.19	0.41	
7-3-5	0.7	0.85	0.85	0.73	0.94	★
7-3-6	1.17	1.11	1.12	0.95	1.25	
7-6-1	1.63	1.23	1.24	1.13	1.34	★

Publication: Sagar Chaki, Joseph Andrew Giampapa, David Kyle, John P. Lehoczky: Optimizing Robotic Team Performance with Probabilistic Model Checking. SIMPAR 2014: 134-145

Team: HCCPS

SEI team members

- Bjorn Andersson, Ph.D.
- Sagar Chaki (co-lead), Ph.D.
- Dionisio de Niz (co-lead) , Ph.D.
- Joseph Giampapa, M.S.
- Arie Gurfinkel, Ph.D.
- John Hudak, M.S.
- Mark Klein, M.S.
- Gabriel Moreno, M.S.
- Lutz Wrage, M.S.

Prior results

- FY11,FY12,FY13 HCCPS line
- FY11,FY12 LENS

Collaborators

- Prof. Marsha Chechik, Univ. of Toronto
- Prof. Ed Clarke, CMU/CS
- Prof. Lui Sha, UIUC
- Prof. John Lehoczky, CMU/Stat
- Prof. Raj Rajkumar, CMU/ECE
- Prof. Anthony Rowe, CMU/ECE
- Prof. Paul Scerri, CMU/RI
- Prof. Natasha Sharygina, Univ. of Lugano
- Prof. Ofer Strichman, Technion, Israel
- Prof. Paulo Tabuada, UCLA

Engaged Stakeholders

- LMCO Russell Kegley, Model problem
- LMCO Jonathan Preston, Model problem

Contact Information Slide Format

Dionisio de Niz
Senior MTS
SSD/CSC
Telephone: +1 412-268-9002
Email: dionisio@sei.cmu.edu

Web
www.sei.cmu.edu
www.sei.cmu.edu/contact.cfm

U.S. Mail
Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

Customer Relations
Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257